CLT: notes on answers

You were asked to write a Matlab/Octave or Python program to demonstrate Central Limit behaviour.

Here's a quick Matlab/Octave solution:

```
N = 1e5;
K = 20;
xx = sum(rand(N, K), 2); % Nx1 array, each element is sum of K uniforms
mu = mean(xx);
vv = var(xx);

% Then you could do the simple histogram comparison we've already seen.
clf; hold on;
hist(xx, 100);
[cc, bin_centres] = hist(xx, 100);
pdf = exp(-0.5*(bin_centres - mu).^2 / vv) / sqrt(2*pi*vv);
bin_width = bin_centres(2) - bin_centres(1);
predicted_bin_heights = pdf * N * bin_width;
plot(bin_centres, predicted_bin_heights, '-r', 'linewidth', 3);
```

I've fitted the mean and variance to the samples. We could derive the mean and variance of the distribution analytically in this case. However, using an empirical fit lets us change the generating process easily. (See below.)

Here's the equivalent first few lines in Python:

```
N = int(1e5)
K = 20
xx = np.sum(np.random.rand(N, K), 1)
mu = xx.mean()  # or np.mean(xx)
vv = xx.var()
# ... and you've seen how to do the plotting already
```

For keen students: a Q-Q plot would probably be a better way to compare the distributions, as you can see what's going on in the tails better.

The uniform distribution converges very quickly. You could generate samples in many other ways however. Try this Matlab/Octave example:

```
xx = sum(-log(rand(N, K)), 2);
```

Here we transform the uniform samples before adding them up. Using the negative log transformation corresponds to adding up K samples from an exponential distribution. Now the histogram doesn't match a Gaussian very well. The Central Limit Theorem still applies, but $K = 20$ isn't big enough to get close to convergence. As you increase K, the histogram gets gets closer and closer to a Gaussian bell-curve.

If you were to sample values from even heavier-tailed distributions, with infinite variance, the CLT would no longer apply. Keen students could read about examples here:
`https://en.wikipedia.org/wiki/Stable_distribution`